MILESTONE REPORT: Item 0011, Month 12 (M12)

Final Report, Month 12 (M12): "Illustration Watermarking for Digital Images: An Investigation of Hierarchical Signal Inheritances for Nested Object-based Embedding."

1. Cover Sheet

Principal Investigator's name: Prof. Jana Dittmann

Institution's name: Otto-von-Guericke University of Magdeburg

Institution's address: ITI Research Group on Multimedia and Security, Postfach 4120, 39016 Magdeburg, Germany

Grant number: FA8655-06-1-3019

2. Objectives

Objectives of this research effort are to advance the basic research in object-based annotation watermarking to address the problem for Nested Object-based Embedding for hierarchical object compositions in images. Here it is necessary that each annotation does not interfere with any other annotation within the image and furthermore, it is also desirable that the relationship between annotations can directly be expressed by the structure of the different watermarks. As per the proposal, the concept of addressing these research challenges has been structured in two main research tasks:

A) Hyperlink-Graph-Concept: Formal model for representing objects and information in hierarchical structures

and

B) Formal requirements and general design approach for signal-level watermark inheritance

Research of the first 3 months addressed mainly conceptional work for task A), which includes

- Demonstrator System Design
- Ontological Model
- Watermarking Algorithm Evaluation
- Implementation of basic parts of the first demonstrator
- Conceptional Modeling of Hierarchy-Preserving Codes

Referring to task B), months 4-12 addressed the realization of signal-based inheritance, i.e. to transfer the object hierarchy information into the watermark signal. A prototype software application was implemented, that is appended as annexes D1 (program code), D2 (source code) and B (user manual) to this report (on DVD). To estimate the achievements of the new methods, extensive evaluations have been performed; the results of these tests are discussed in chapter 4 of this report. Test results and also included in annex D3 (spreadsheet) and D4 (complete log files of the tests).

3. Status of effort

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
23-02-2007	Final Report	8 December 2005 - 08-Dec-06

4. TITLE AND SUBTITLE

Illustration Watermarking for Digital Images: An Investigation of Hierarchical Signal Inheritances for Nested Object-based Embedding

5a. CONTRACT NUMBER
FA8655-06-1-3019

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Dr. Jana Dittmann

5d. PROJECT NUMBER

5d. TASK NUMBER

5e. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Otto-von-Guericke University of Magdeburg
Universitätsplatz 2
Magdeburg 39106
Germany

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

EOARD
PSC 821 BOX 14
FPO AE 09421-0014

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Grant 06-3019

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Objective of this research in object-based annotation watermarking is to address the problem of Nested Object-based Embedding for hierarchical object compositions in images. Here it is necessary that each annotation does not interfere with any other annotation within the image and furthermore, it is also desirable that the relationship between annotations can directly be expressed by the structure of the different watermarks. The concept of addressing these research challenges has been structured in two main research tasks: A) Hyperlink-Graph-Concept: Formal model for representing objects and information in hierarchical structures; and B) Formal requirements and general design approach for signal-level watermark inheritance.

15. SUBJECT TERMS

EOARD, Steganography, Image Fusion, Data Mining, Image Processing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18, NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			P. B. Losiewicz EOARD/IFT
UNCLAS	UNCLAS	UNCLAS	UL	20	19b. TELEPHONE NUMBER *(Include area code)*

The project started January 1st, 2006. During the first month, the work focussed on prototype design issues, which includes issues of development platform, plug-in interface design, ontological model for object hierarchies and user interfaces. All these aspects could be finalized (see section accomplishments) and implementation of the prototype started in the first period.
As the basis for implementing the Hyperlink-Graph-Concept, a model for a Hierarchy-preserving codebook has been developed and finalized. This scheme requires an underlying watermarking method for data embedding, for which an appropriate technique had to be identified by evaluating watermarking schemes. Here, design goals have been studied and two reference methods have been chosen: For robust spatial embedding the luminance block watermark by J. Dittmann / J. Fridrich has been selected and for high capacity embedding the WET paper code algorithm by J. Fridrich. Additionally to the two selected reference methods, a novel scheme, denoted as Hierarchical DDD (Dual Domain DFT) algorithm has been developed within this effort. The latter has been done with respect to the formal requirements and the general design approach for signal-level watermark inheritance. By end of M12, all three mentioned embedding schemes have been implemented as dynamic library versions, integrated in prototypical system and practically been evaluated.

4. Accomplishments/New Findings

This chapter will provide comprehensive demonstrations of the accomplishments and new findings acquired during this project. During the initial stage of the project, this mainly affects the three areas *System Design & Implementation*, *Watermarking Algorithm Evaluation* and *Hierarchy-preserving codes*. These aspects will be discussed individually in the first part of this chapter.

Subsequently, in the concluding sections of this chapter, more recent findings like our approach to use *Wet Paper Codes for Nested Object Watermarking* or our novel *Hierarchical Dual-Domain-DFT Watermarking* scheme will be introduced as well as the *Experimental evaluation setup and results* of our practical tests.

4.1. System Design & Implementation

Design goals for the Hyperlink-Graph-Concept demonstrator need to address mainly the aspects of development platform, ontological syntax to represent object hierarchy, software architecture and user interfaces.
Regarding the development platform, it has been decided to make use of a rapid prototyping integrated development environment (Borland Delphi IDE, [1]), because it provides numerous high-level image processing functions (e.g. format conversion, re-scaling et cetera), as well as visual software components that implement user interface controls efficiently. On the other end, the concept of Delphi IDE includes a low-level, Dynamic Link Library (DLL) interfacing concept, allowing to include functionality of software modules implemented in literally any other programming language, at run-time. Thus, this concept allows time-efficient implementation of the user interface part of the demonstrator, in a way that the signal-level modules that will be developed along with our fundamental work can be implemented independently of the user interface part (plug-in concept).
In order to allow interaction between the Watermarking Editor (WM Editor, User Interface part) and the plug-in watermarking algorithms, a program interface and protocol has been developed, which allows control of the embedding and retrieval processes by the WM Editor.

The embedding protocol is illustrated in Figure 1 and consists of five base functions:

1) **registration** of the specific WM algorithm with the editor: this is required, because multiple algorithms shall be supported within one WM editor,

2) **capacity validation**: editor transmits message and spatial area to WM algorithm, WM algorithm reports back if capacity of spatial area is sufficient to embed message,
3) **message embedding**: editor again transmits message and spatial area to WM algorithm, WM algorithm reports back if embedding was performed successful or not.

These functions are identified by arrows between the WM Editor and Algorithm and vice versa in Figure 1.

Figure 2 illustrates the retrieval protocol. Because the watermarking schemes used in context of this research are all of blind natures and we expect multiple watermarks in each image, the WM editor sequentially requests retrieval of watermark messages and the WM algorithm will either return the message found or an end-of sequence message, if no more messages are found. Consequently, the WM editor actively polls retrieval message by message and may thus collect the entire set of watermarks in an image.

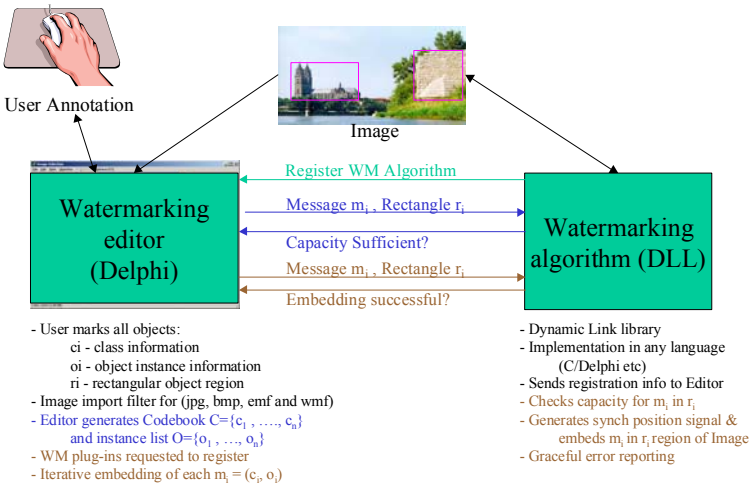

Figure 1 Demonstrator System Design: Embedding Architecture & Protocol

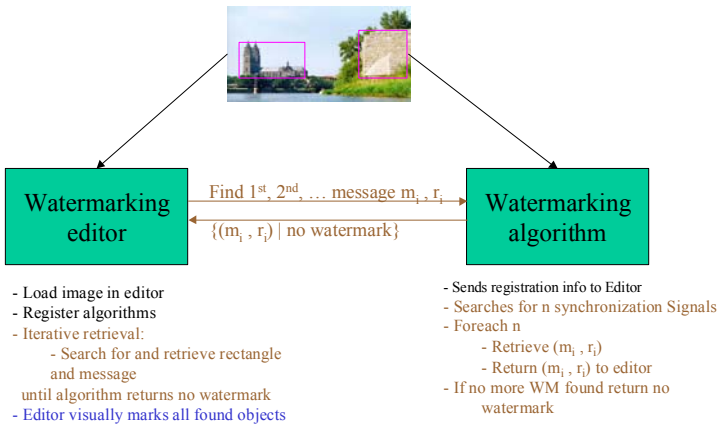

Figure 2 Demonstrator System Design: Retrieval Architecture & Protocol

For the representation of the object hierarchy, a formal structure is required, as well as an exemplary ontological database. To assist the user at finding a name for an object and avoid the usage of different alternate names or spellings for same objects a lexical ontology required. WordNet is free and an open source ontology from the Princeton University ([2], [3]), provid-

ing a well-defined data structure and a large database consisting of approx. 155.000 English nouns, verbs, adjectives and adverbs ordered by synonyms. For our research, it has been that the noun parts of the WorldNet project are well suited to name objects. Additionally, for each word and its synonyms relations to other words are given, e.g. antonyms (words with opposite meaning), hypernyms/hyponyms (kind-of-relationships) and holonyms/meronyms (part-of-relationships). In our implementation we refer to the holonym-meronym relationships in WorldNet, where X is a meronym of Y and Y a holonym of X if X is a part of Y. This concept has been integrated in the system design, and a holonym-meronym object browser is already implemented in the demonstrator. Figure 3 illustrates an example for a relationship in the object browser between objects “exterior door” (holonym) and “doorknob” (meronym).

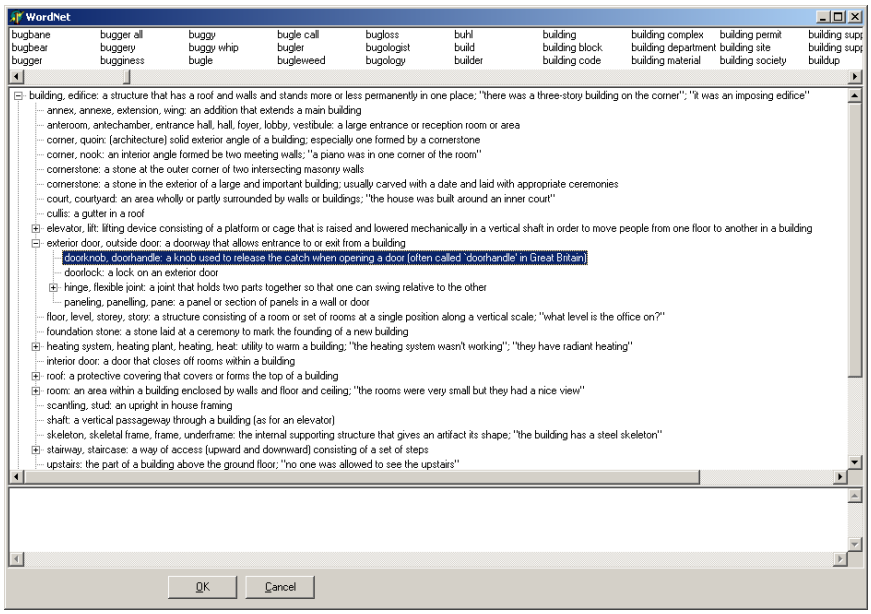

Figure 3 WorldNet object browser example: holonym-meronym relationship between “exterior door” and “doorknob”

For the support of the user in selecting appropriate regions for watermark embedding, the Editor part of the demonstrator shall be equipped with an **automated contour detection function**. To this end, BlobContours, an algorithm that has been introduced by our research group in earlier work ([4]) has shown quite good efficiency with respect to contour detection results and computational performance. The user interface part shall thus support selection of rectangular areas as well as an automatic contour detection. Due to the constraints of existing watermarking algorithm with respect to embedding region (see next section), the result of contour detection will be reduced to a bounding-box rectangle for the reference algorithms. However the concept is open to future watermarking schemes that may support polygonal embedding shapes.

4.2. Watermarking Algorithm Evaluation

For evaluation of watermarking schemes that are appropriate for representing hierarchical objects, it is necessary to discuss conceptual constraints as well as design goals for the algorithm.

Conceptual constraints exist due to the fact that illustration watermarking implies embedding of multiple watermarks in one single image, i.e. there exists a limitation in spatial area available.

Related work has addressed region-based embedding of payload, with the goal to embed data in regions that are less vulnerable to image modifications ([5]), however schemes for representing hierarchical information in context of hierarchical annotations have not been studied so far. The problem becomes particularly complex for our case of hierarchical objects with functional-spatial relations, because this naturally implied overlaps of the spatial regions within one object hierarchy (e.g. a door as part of a building object will obviously be located within the shape of the parent object). Furthermore, another degree of freedom is the shape contour of the spatial area. While the user may define the spatial boundaries of objects by contours, for example using an automatic contour recognition such as BlobContours ([4]), in terms of polygons, to date, watermarking algorithms typically embed the message pseudo-randomly across an entire image in order to preserve transparency and capacity. However, the specific requirement of annotation watermarking, i.e. the arbitrary selection of the embedding area to a user-defined shape and the implicate spatial relation of the watermark to its location in the image may thus leads to insufficient capacity and limitation in spreading the information across the image.

Design goals for digital watermarks are the three conflictive aspects: Robustness, Capacity and Transparency. Optimization towards one of these aspects always implies trade-off for the other goals.

For illustration watermarks, apparently one **main goal of robustness is robustness against cropping**. Particular for object hierarchies, the goal is to identify hierarchical relations, even if only part of the original image is available during retrieval. If for example, an object "Exterior Door" as part of "building" has been cropped from an annotated, larger image, the goal is to be able to identify the object class type and the hierarchy (i.e. the fact that the door is part of a larger hierarchy), even from the remaining image. Robustness against other forms of modifications, such as scaling or geometrical attacks are relatively unimportant, because of the application scenario, where we do not expect targeted attacks. Robustness against lossy compression is an interesting aspect in application scenarios where memory limitations are expected.
The goal of Capacity is of interest for illustration watermarks because of the before mentioned spatial limitation of the embedding area. This research will thus comparatively consider both low-capacity and high-capacity schemes, whereas Transparence aspects are of subordinate importance.
In the first stage of the project, our evaluation identified two watermarking schemes for the further elaboration.
The first method has been introduced by J. Dittmann ([6]) and is based on modulation of the luminance signal in 8x8 patterns, as suggested by J. Fridrich ([7]). The method promises robustness against cropping and lossy compression, at a relatively low capacity.
The second method, Wet paper codes, as introduced by J. Fridrich et al. ([8]), promise relatively high capacity, but due to its steganographic character only very limited robustness against cropping.
Both algorithms have been extended by the following features:

- Generation and detection of **synchronization patterns**
- Spatial limitation to the **embedding area** boundaries

To date, the above mentioned features have been implemented for the two embedding schemes (Block-Luminance and Wet Paper Codes) as run-time libraries (Windows™ DLL), which embed payload data in a generic way (i.e. input parameters are coordinates of a rectangular area, embedding strength, cover image and payload data) in a given image. Within this

project, these software modules have been used to study Annotation Watermarking based on the Hierarchical Tree Codebooks (see section 4.3), however; the usability of the DLLs is not limited to other applications in the future.
Due to the before mentioned requirement of synchronization pattern & the design constraints of spatial embedding, we have decided to consider rectangular areas (marked with a mouse or a pen device or with an automated contour detection method such as BlobContours) for the embedding contours for the demonstrator. Further, to address the problem of user-specific size for the areas, the following embedding protocol has been developed, based on 8x8 pixel blocks for the luminance-watermarking algorithm:

- Embedding of message m_i in a rectangular area of an image defined by upper left corner (x_i , y_i) and width and height w_i and h_i respectively. All data except the synchronization pattern is embedded with triple redundancy.
  - Generate a 15-bit synchronization pattern *synch*
  - Embed *synch* in the top left 15 blocks, starting from (x_i , y_i) in a 8x8 block row from left to right.
  - Embed first half (5 bits) of rectangle width w_i in the 8x8 block row just below synch (i.e starting from (x_i +8, y_i).
  - Embed second half (5 bits) of rectangle width w_i in the 8x8 block row just below synch (i.e starting from (x_i +16, y_i).
  - All subsequent 8x8 block lines will utilize w_i , i.e. $\lfloor w_i / 8 \rfloor$blocks per row, because in retrieval, the true width is known from this protocol step onwards.
  - Embed height (10 bit), message length (16 bit) and message content of m_i .

Figure 4 illustrates an example for this embedding protocol, note that in the first three lines of 8x8 blocks, a minimum rectangle with of 15x8 = 120 pixels is required, whereas the protocol utilizes all $\lfloor w_i / 8 \rfloor$blocks from the fourth row onwards.

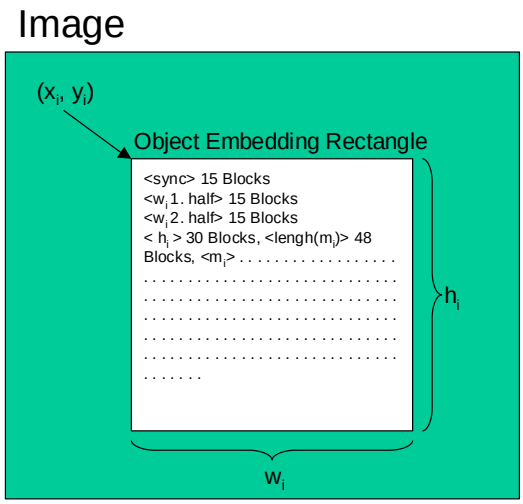

Figure 4 Example for the embedding protocol

Our first experiments have indicated that the synchronization pattern detection is a crucial problem of this protocol, because in our first test image, a number of random patterns with the same bit sequence occurred, thus leading to falsely detected synchronization patterns. Future research will therefore address the optimization of pattern detection and its combination with the hierarchical codebook.

4.3. Hierarchy-preserving codes[1]

The initial question here is how to formalize visual-functional and/or visual-spatial relationship as annotation itself and how to embed it into a watermark.

As suggested in the project proposal, we started the investigation of Hierarchical Trees (HT) with respect to our requirements and analyze further techniques that could be used to map the coherency of marked objects. To this end, we have developed a formal representation in form of trees and a codebook approach to represent such class tree diagrams. Such a diagram can be derived for example for the annotations shown in Figure 5. Here, an exemplary image containing the spatial object annotations for two objects (denoted as b1 and b2) of class type "Building" is shown. In this example, the annotation consists of sub-class objects of type "Window" for each of the buildings (w1.1, w1.2 for b1 and w2.1 to w2.5 for b2), as well as one "Exterior Door" (d1.1 and d2.1). Furthermore, for the left door, its subclasses "Doorlock" and "Doorknob" have been annotated, as well as for the two windows of the left building, the subclasses "Window Frame"(not visible in Figure 5).

Figure 5 Image Example: 2 root objects of class "building" and their sub classes.

For this example, the HT class diagram is shown in Figure 6. It consists of three levels of hierarchy: a root class c_1, having two branches classes $c_{1.1}$ and $c_{1.2}$ and finally the three leaf classes $c_{1.1.1}$, $c_{1.1.2}$, $c_{1.2.1}$. Note that the HT class diagram models solely class relations without addressing the instantiation of these classes into objects. For the example from Figure 5, this implies, that the HT class diagram does not provide an instantiation mechanism for the two building objects b1 and b2 respectively, but rather a model for the class relations between a class of type "Building" and the annotated sub classes.

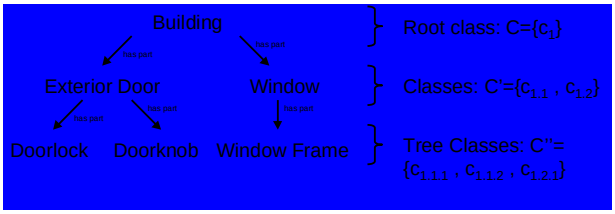

Figure 6 Hierarchical Tree for the object relations from Figure 5.

The HT codebook scheme representing such class hierarchies is a binary code and is generated for any meronym class B of A (i.e. B being part of A) as follows. Let the parent class A of B have a set of child classes, then each child class is sorted by its children count in decreasing order and indexed beginning with the number zero, whereby the order is arbitrary in case of identical children counts. Provided B is the *n*-th child class of A, then the binary code is the recursively created code of class A, concatenated by a sub-code consisting of *n* ones, followed by one zero. Each of the zero symbols then represents the level of the corresponding object in the class tree, whereas the ones represent the child's index. Due to the sorting by the number of class objects in the given annotation, the class code length is locally minimized. For the example in Figure 6, class "Building" is at root level, it can thus be interpreted as the first child of a virtual class, having code "0". The sub class "Exterior door" of "Building", having more children than "Window", becomes the first sorted child (having index 0) and therefore

[1] Note this section has been updated from the version in Milestone Report M3, based on the description in [ViDi2007]

the code "00" (leftmost 0 for the parent class, rightmost for the actual class). "Window", being the second child of "Building" is assigned the code "010" (leftmost 0 again for the parent class, rightmost 10 for the actual class). Their child classes' codes are recursively constructed in the same manner and the resulting codebook for the example from Figure 6 is shown in Table 1.

Class name	Class code	Class name	Class code
Building	0	Exterior Door	00
Window	010	Doorlock	000
Window Frame	0100	Doorknob	0010

Table 1: Exemplary class codes for the image in Figure 7 and the HT diagram in Figure 6.

Generally in this codebook, class codes possess a property, which allows a simple validation of object hierarchy for any two given codes c_1 and c_2, where length of c_1 is less than the length of c_2: c_2 is a meronym of c_1 (and c_1 a holonym of c_2) if and only if the leftmost *length*(c_1) bits of c_2 are identical to c_1. Note that this relation can be validated across any number of class hierarchy levels. Applying this test for the example to the codes presented in Table 1, the code c_2=*0010* unveils directly that class Doorknob is a meronym to Exterior Door (c_1=*00*) as well as to Building (c_1=*0*), but not to Window (c_1=*010*). By using the codebook scheme for representing the class hierarchies and some instantiation mechanism for the objects in annotations, hierarchical object watermarking can be achieved with literally any underlying data embedding scheme. In this effort, the two selected embedding schemes from Watermarking Algorithm Evaluation (see section 4.2), Block-Luminance and Wet Paper Codes, have been implemented into the prototype system, AnnoWaNO.

4.4. Block-Luminance Data Embedding

The first data algorithm, which has been selected for implementation of HT codebook embedding, is the Block-Luminance (BL) method. In extension to the original method from [9], it generates a synchronization pattern, utilized to locate the spatial position of the first embedding. This is required to ensure robustness against cropping in blind detection scenarios, where an exhaustive search for synchronization symbols is necessary, if cropping borders are not multitudes of the block size.

The concept of annotation watermarking requires the approach of synchronization, because we want to ensure robustness against cropping in blind detection scenarios, where an exhaustive search for synchronization is required. The synchronization pattern in the reference implementation is a 32-bit binary sequence. For our evaluation, we have considered patterns represented by the following hexadecimal numbers: *$00000000*, *$FFFFFFFF*, *$55555555*, *$AAAAAAAA*, *$B4B4B4B4*, *$40014001* and *$11111111*. Further details of the BL embedding scheme, including an embedding protocol for annotation watermarking, can be seen from [10]. Since in pre-evaluation of the patterns, the pattern *$40014001* has shown a good performance trade-off with respect to transparency versus robustness (detectability), the further evaluation of the BL scheme has been limited to this setting. The entire test protocols for the remaining settings are included in Annex D4 to this report.

4.5. Wet Paper Codes for Nested Object Watermarking

Our implementation is based on the theoretical approach, suggested by Fridrich et al. in [11], [12] and [13]. Details of how to use Wet Paper Codes (WPC) for digital watermarking can be found in the original contributions, whereas we will focus on a very brief summary of the general concept and the simplifications which we have chosen for our implementation.

In a very general view, the coding of WPC are computed from solutions of a linear equation system, $H \cdot v = m - D \cdot b$. Variables in this equation are three vectors and two matrices. b denotes a binary column vector, defining a set of indices $C \in \{0, 1, ..., n-1\}$, $|C| = k$ of those bits that can be modified to embed a message. m is the $q \times 1$ binary message vector and v is an

unknown $k \times 1$ binary vector. Matrix D denotes a pseudo-random binary matrix of dimensions $q \times n$ generated by a shared secret key, whereas H is a binary $q \times k$ matrix consisting of those columns of D corresponding to indices in set C. With the equivalence of $v = b' - b$, the embedder of a message generates the code by modifying each of the C positions in b, b_j, $j \in C$, so that the modified binary column vector b' satisfies $D \cdot b' = m$. Using the same shared matrix D, the decoder can retrieve the message m in an analog manner.
A very detailed description of WPC and their application for stenography is provided in [11], in this subsection, we will refrain from discussing further details of this scheme and focus on the specifics of our implementation for a comparative evaluation. Our deviations from the original approach are threefold:

1. Generation of the matrix D: In our implementation the matrix D is generated with a given fixed size ($q=8$, $n=40$). This means that the message is adapted to the matrix size and divided into partial messages as a function of q and n.
2. Permutation of the vector v: In our implementation we try to find solutions of the linear system of equations $H \cdot v = m - D \cdot b$, by permutating vector v. In case that a solution is not found for a given D, matrix D must be generated again, based on a another key k' ($k' \neq k$) and a solution of the linear system of equations is computed again, based on the permutation of vector v. In [11], this problem is by modifying the parameters q and n of matrix D are modified, until a solution is found.
3. Algorithm for the solution of large linear system of equations: the most complex issue of the Wet Paper Code approach is the search for a solution of the system of equations equations $H \cdot v = m - D \cdot b$. In our implementation we use a very baseline method for this, the Gauss's algorithm, which works properly only for small system of equations. Although small dimensions of the (secret) matrix D implies some security deficits, we have chosen this limitation because security is not the main goal of annotation watermarking and performance issues are more significant for us.

As a steganographic channel to embed the coded data, we have chosen the blue channel Least Significant Bit (LSB), due to an expected high transparency. Further, the adaptation of WPC for annotation watermarking required the definition of an embedding protocol, which we designed with the following main properties. The algorithm was adapted to embed a message in an object region, selected as a rectangular part of the cover image. Further, a synchronization pattern is embedded in the selected region and provides information about the position, at which position the actual annotation watermark has been embedded. Finally, the algorithm was adapted to the interface requirements of the illustration watermarking tool for nested objects: Annotation Watermarking for Nested Objects (AnnoWaNO).

4.6. Hierarchical Dual-Domain-DFT Watermarking

In addition to the Hyperlink-Graph Model and the resulting HT codebook approach described in sections 4.2 to 4.5, the second main contribution of this project was the study how to perform signal-level watermark inheritance rather than modeling hierarchies into codes, which are then embedded using an arbitrary embedding scheme such as Block-Luminance or Wet Paper codes. To this end, we were able to conceptionally design a new approach, denoted as Hierarchical Dual-Domain-DFT (DDD) Watermarking, integrate this scheme into the AnnoWaNO prototype system and to evaluate our new approach in comparison to the previous ones (see section 4.7).
Our new developed approach of Hierarchical DDD Watermarking is based on two main concepts. Firstly, the class hierarchy information is separately embedded (i.e. in a different domain) from any other object instantiation data whereby the class hierarchy is synchronized by means of a presents bit in the object instantiation. Secondly, the Hierarchical DDD scheme is

designed in such way, that object hierarchy relations are represented by inherited properties between the embedding signals intrinsically, i.e. without the need of having annotation-specific code books as suggested for example by the Hierarchical Graph Concept (HGC). Our new concept follows the idea of spread spectrum watermarking based on modulation of magnitude and phase in the DFT (Digital Fourier Transformation) domain.
DFT methods have been reported to be capable to generate watermarks with a relatively good trade-off between transparency and robustness. Although initial work has been suggested relatively long ago for spread spectrum image watermarking ([14], [15]), still novel DFT methods have been suggested more recently, for example for multiple watermark embedding ([16]). As compared to other approaches, we separate class hierarchy information from instantiation data (e.g. the actual watermark payload). We do so by firstly assigning sub frequency embedding bands to each class and modulation of their magnitudes such that magnitude relations of the DFT Coefficients intrinsically inherit the class hierarchy. Secondly, our approach utilizes phase modulation in the same DFT domain. The methods for this new dual-domain embedding are described in more detail in the following paragraphs.

Embedding of Class Hierarchy

As a first step in the embedding process, the annotation areas of the image are transformed into the DFT domain. In our scheme, this is based on blocks of $n \times n$ pixels, thus resulting in n^2 /2 coefficients for magnitude and phase respectively, representing the positive frequency shares in the original signals from $f=0$ to the Nyquist frequency f_{nyc}. Secondly, the class hierarchy path for the current object is generated by enumerating the nodes of the hierarchy tree from the actual class node towards the root starting with zero, so every node in the path has a unique number. Therefore the hierarchy path of a watermark class is an ordered list of integers, which is denoted as queue data structure *hier* in the further discussions of this algorithm. Each of the queue components contains an individual offset value of a class node within the list of frequency bands l and relative to f_{nyc}. The head component, denoted as *hier[0]*, therefore represents the offset of the actual class node itself and the last element the id of the root parent. Removal of the head component leads to a queue length reduced by 1 and the previously second object becoming the new head component, *hier[0]*.
The hierarchy path is embedded in some of the magnitudes of this data frequency band, whereby the data frequency band is limited by a system parameter cut-off frequency f_{cutoff} , as well as the Nyquist frequency f_{nyq}, where f_{cutoff} is in the range of $[0, \ldots, f_{nyq}]$. The length l of the frequency band is therefore defined as $l = f_{nyq} - f_{cutoff} + 1$, i.e. the magnitudes of the l highest frequencies in the spectrum of a given block are used to embed the class hierarchy. Consequently, l is also the upper bound for the number of hierarchy classes that can be represented by the watermark. Another system parameter, hierarchy depth d, defines how many nodes of the hierarchy path are embedded above the noise threshold, whereby all preexisting magnitudes of the data frequency band are considered as noise. The noise threshold is therefore the maximum magnitude of all components of the data frequency band l of every embedding block. A third parameter, the embedding strength factor s defines, the maximum ratio between the signal (hierarchy path node) and the noise threshold.
During the embedding, the algorithm iterates in the hierarchy path from the actual node, *hier[0]*, towards the root with a maximum depth of d. Hereby the resulting embedding strength factor decreases for every parent, relatively from an actual node, whereas the child's id is embedded with the maximum factor of s the d-th parent's id is embedded with a factor of 1.
This is achieved by reducing every parent's embedding strength to its child's strength s divided by the d-th root of s. In pseudo-code notation, with *M[f]* denoting the magnitude function of a DFT coefficient related to frequency f, and *effstrength* the value of the actual effective embedding strength in each iteration, the algorithm can be described as follows:

Step 1: `effstrength := max{M[fcutoff], …, M[fnyq]} · s`
Step 2: `M[fcutoff + hier[0]] := effstrength`
Step 3: `effstrength := effstrength / (d · s^(½))`
Step 4: Remove head component `hier[0]` from `hier`.
Step 5: If `hier` is not empty go to Step 2, otherwise finished.

Figure 8 illustrates the effect of this embedding algorithm for an exemplary magnitude distribution of DFT coefficients and the parameters *s=20*, *d=3* and *l=64*. The index values of *hier[0]*, …, *hier[3]* are chosen arbitrarily for this example and consequently, the frequency coefficient related to *fcutoff+hier[0]* receives the highest embedding energy (see highest of the four stripes columns in the left-hand frequency bands). The coefficient related to *hier[3]* receives the lowest embedding energy, which is equal to the maximum magnitude above all original frequencies in the embedding frequency band (see striped column having the least height).

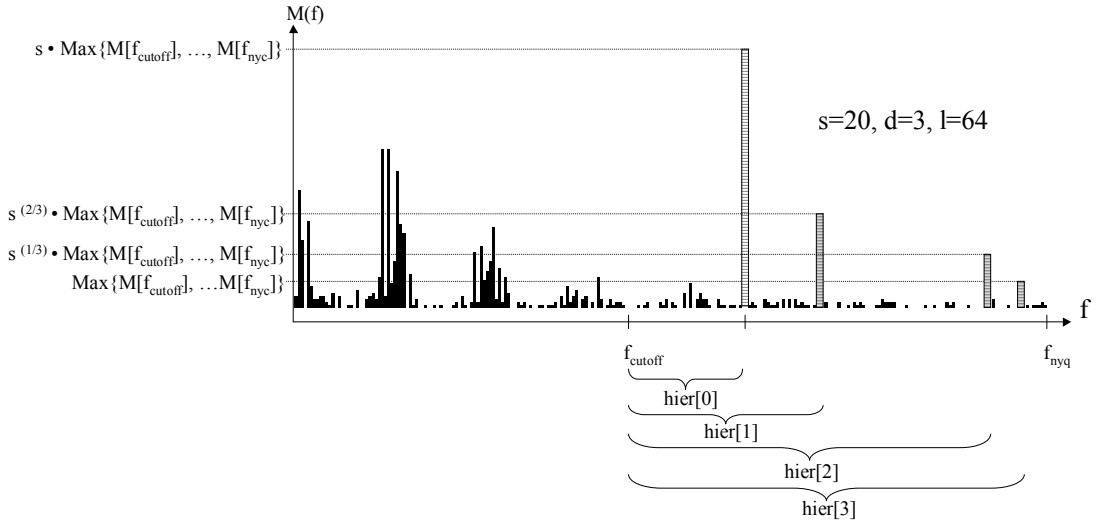

Figure 8 Example for the modulation of DFT magnitude coefficients for s=20, d=3 and l=64

Retrieval of Class Hierarchy

The retrieval starts with the search for a presence bit in the object instantiation (see description in the following paragraph regarding object instantiation) with an exhaustive search strategy. Afterwards, the retrieval is performed for all identified hierarchy regions on all watermarked blocks of each of the annotated object regions. For every watermarked block, except the ones overwritten by another watermark, the magnitudes in the before mentioned data frequency band $f_{cutoff}, \ldots, f_{nyq}$ are normalized to values between *0* and *1*. The class hierarchy is then restored by iterating through all frequencies *f* in the frequency band of length *l* in the following scheme:

For every frequency *f*, the mean μ_f and standard deviation σ_f above all blocks of the image (or part-of) are calculated. If μ_f is zero, the frequency is ignored, otherwise the possible hierarchy level *posslvl* is the negative of the $\sqrt[d]{s}$-*th* logarithm of μ_f. The ratio $ratio_f$ between *posslvl* and its nearest integer hierarchy level *nearlevel* is calculated by subtracting both and scaling them back to linear scale. The magnitudes of *f* in each of the watermarked blocks are assumed to represent a hierarchy level equal to *nearlevel* if μ_f as well as σ_f are in an acceptable range. The combined valid range is between $ratio_f - \sigma_f > 1 - \theta$ and $ratio_f + \sigma_f < 1 + \theta$, for our experiments, we have intuitively set the value for θ to 0.25.

Embedding and Retrieval of Object Instantiation

The embedding, as well as the retrieval of the message bits is done on *bandcount* = 4+1 frequency sub-bands located between the cutoff and the Nyquist frequency. Each sub-band consists therefore of *bandlen* = (f_{nyq} – f_{cutoff}) / *bandcount* frequencies. For a sub-band 0<=*b*<*bandcount* the upper frequency is $f_{upper,b}$ = (f_{nyq} – 1) – *bandlen* * *bandcount* and the lower $f_{lower,b}$ = $f_{upper,b}$ – *bandlen* + 1. Each data bit is embedded in the phases of one sub-band.
One of the sub-bands (in our case the first one, *b=0*) carries a watermark presence bit rather than message payload. This presence bit has a fixed value of 1 to mark an image block as watermarked. The other 4 sub-bands carry the actual payload *M'*, whereby *M'* is derived by the actual message *M* by preceded by the message length as the retriever must know how far to read. Both are finally coded with a (255,223) Reed-Solomon-code (RS) for error detection and correction, i.e. *k=223* 8-bit data symbols plus 32 8-bit parity symbols are coded into *n=255*-symbol blocks, allowing to correct up to 16 symbol errors per byte block.
The embedding for the presence and the message bits is done as follows. A binary 0 is represented by the phase angle $\varphi_0 = -0{,}5\pi$ and a binary 1 by $\varphi_1 = +0{,}5\pi$. Every phase in one sub-band is set to either φ_0 or φ_1, additionally added with a pseudo-random phase angle. This is not just done for security as the PRNG sequence depends on a key. To allow a correct retrieval of the embedded message, the PRNG must not only be initialized with the same key but the watermarked frequencies must read in the same order as they were changed during embedding. This further leads to the necessity of the correct read order of the sub-bands and the blocks. An incorrect size or shape of the watermark or another watermark's blocks overlapping the current watermark's blocks disturbs this order. Inversely due to the PRNG-based phase rotation the watermark size and shape can be found by a trial-and-error method as well as overwritten blocks.
For the actual retrieval, our scheme takes a probabilistic approach, whereby for retrieval of each bit out of one sub-band *b*, the pseudo-random phase angle is subtracted from each phase Φ_b in the current sub-bands. From these phases the magnitude-weighted mean μ_b and standard deviation σ_b is calculated. The returned value is not a binary one but a real value between [0…1]. This value is computed out of the probabilities that Φ_b contains an embedded 0 or 1*: res = (P(b=0) + P(b=1) – 1) / 2*.
Assuming the phases in Φ_b follow a normal distribution, *P(b =0)* is the probability that a $N(\Phi_b; \sigma_b \mu_b; \sigma_b)$ normal distribution takes values between $(\varphi_0 - 0.5)$ and $(\varphi_0 + 0.5)$:

$$P(b=0) = P(\varphi_0 - 0.5 < \Phi_b < \varphi_0 + 0.5) = P(\Phi_b < \varphi_0 + 0.5) - P(\Phi_b < \varphi_0 - 0.5)\ ,$$

with:

$P(\Phi_b < x) = (1 + erf((x - \mu_b) / (\sigma_b \sqrt{2}))) / 2$ and *erf* being the Gauss error function.

P(b =1) is analogously retrieved by replacing φ_0 with φ_1. In our implementation, in the first band (*b = 0*) the presence bit is assumed to be detected if *res* >= 0.9. For the remaining bands (*b* > 1) it is assumed that a message bit is one if *res* > 0.5 and zero if *res* < 0.5. The latter criteria have a less strict threshold because of their RS error correction code.

4.7. Experimental evaluation setup and results

Our test goals are twofold. Firstly we evaluate fixed object annotations (with fixed capacity requirements for 18 annotations) with respect to the transparency by an objective measurement and its robustness to compression with and without error corrections (goal A). Secondly the objective transparency and robustness to compression and cropping are evaluated for individual (manually performed) object annotations by determining also the impact of error corrections (goal B). For both of these goals we determined the following transparency and robustness measurements for all three tested algorithms: The PSNR between the original image

and the embedding signal, the Bit Error Rate (BER) for raw retrieved annotations (RBER), as well as the error corrected annotation (CBER). Furthermore, we have measured the overall watermark detection rate (successful retrieval of presents bit and reconstruction of correct hierarchy), that have been determined directly from each of the watermarked images and after JPEG compressions of 100% 90%, 75%, 50% and 25% quality grade respectively.

For the first part of our experiments we used an image database consisting of 108 images with the required capacity for our test annotations (minimum width and height of the spatial annotation area), a selection from 306 images from the Watermark Evaluation Testbed (WET, [KOGD2004]). With respect to the reference object hierarchy definition, the same exemplary hierarchy as introduced in [10] has been applied. It consists of three level nested object annotations, structured in 3 classes with 9 instances at the root level, 4 classes with 7 instances at the intermediate level and 1 class with 2 instances at the leaf level. This resulted in a total of 18 object annotations and therefore watermarks. In our further discussions, we also denote this as *Test Set A*.

In the second test setup, also referred to as *Test Set B*, where we intended to be more related to practice, we ran an evaluation using another set of images, which have individually been annotated by a human user (for example, see Figure 9) using the AnnoWaNO software. From another image database (containing 91 royalty free test photographs taken during the project, at least 5.9 MPixels each) we selected 15 photos as test-set for this experiment, for each of which we created individual annotations. These 15 reference-annotations contain at least 9 and at most 11 hierarchically nested objects with an overall depth between two and three levels. As a separate step in this test-setup, we determined how individual cropping of annotated images influences on the detection rates. For each reference image, a human user individually chose and cropped an area of his interest, which contained between 21% and 58% (39% on the average) of the original image area. We used the same reference annotations and algorithm parameterizations as above (however, we didn't consider additional JPEG-compression this time to keep the number of test cases feasible) and cropped the selected areas from each of the output images.

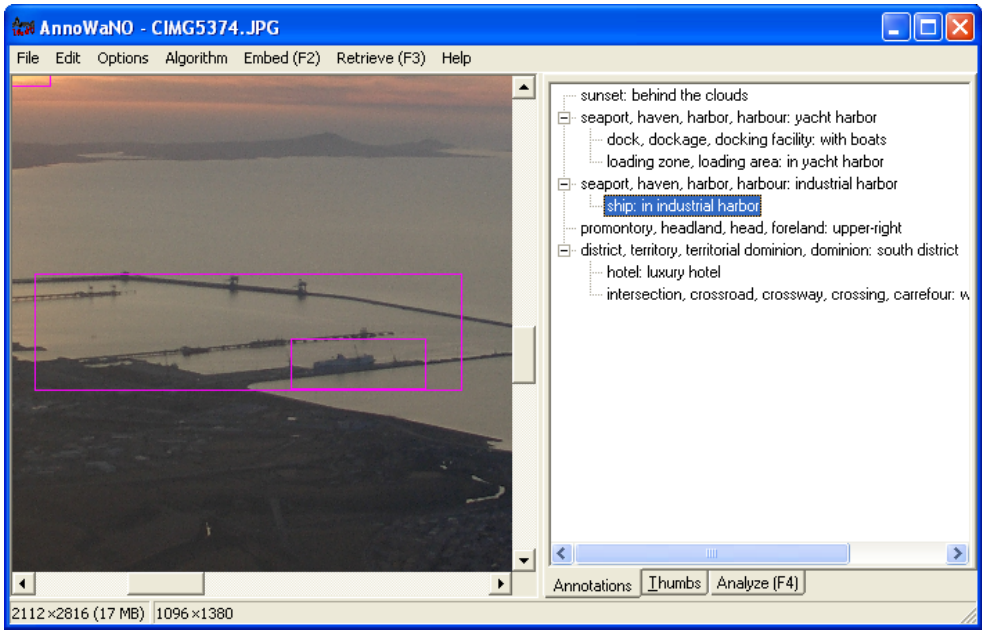

Figure 9 The AnnoWaNo application during creation of a reference hierarchy: seaport - ship

During Test Set A, as parameterization for the different embedding schemes we used an embedding strength of s=3 for the Block Luminance algorithm (that uses a mid frequency sync block and $40014001 as sync sequence by default) and s=5 for the Hierarchical DDD algorithm. The WET Paper Code algorithm needs no further parameterization.

In Test Set B, we ran all possible combinations of four different embedding strengths s (1, 3, 5 and 10 for the Block Luminance algorithm and 1, 5, 10 and 20 for the Hierarchical DDD algorithm) and other algorithm-dependent settings (sync blocks of low, mid and high frequency and sync-sequences of $00000000, $FFFFFFFF, $55555555, $AAAAAAAA, $B4B4B4B4, $40014001 and $11111111 for the Block Luminance algorithm).
In both experiments we determined the following measurements for all tested algorithms. The PSNR between the original image and the embedding signal, the Bit Error Rate (BER) and the watermark detection rate, that have been determined directly from each of the watermarked images and after JPEG compressions of 100% 90%, 75%, 50% and 25% quality grade respectively.

Results

The measured data presented in this section for both Test Sets these refers to the parameterization of s=3 for the Block Luminance algorithm (at a medium sync block frequency and sync sequences of $40014001) and s=5 for the Hierarchical DDD algorithm. These settings have shown a good performance trade-off with respect to transparency versus robustness throughout our tests. For the complete and more detailed test results see Annexes D3 (structured excel sheet) and D4 (the plain source log files).
Table 2 (Test Set A) and Table 3 (Test Set B) show the corresponding results from both test setups. In the top rows, averages of Bit Error Rate, Watermark Detection Rate and PSNR for all images are shown. Further, minimum, maximum and standard deviation are given in the lower three rows. The compression rate is given in terms of JPEG quality factor, denoted as J<x>, whereby expression <x> stands for a factor between 1 and 100 in percent. Note that PSNR measurements have not been performed for the Wet Paper code algorithm after JPEG compression, because compression at any rate resulted in 100% detection error rates

Algorithm:	Block-Luminance						DDD						Wet Paper Code	
Compression after Emb.	Raw	J100	J90	J75	J50	J25	Raw	J100	J90	J75	J50	J25	Raw	J100
Average RBER [%]	0.06	0.07	0.09	0.23	0.59	2.51	0.00	0.00	2.98	22.22	36.76	47.58	n/c	n/c
WM Detection Rate [%]	99.1	99.1	98.8	98.2	95.8	87.4	100.0	99.9	16.0	1.4	0.0	0.0	100	0
Average PSNR [dB]	47.16	45.94	44.03	40.13	36.96	34.34	44.9	44.0	43.0	39.9	37.2	34.5	87,82	n/c
Minimum PSNR [dB]	42	41	40	36	31	23	34.95	34.55	34.22	34.03	30.79	22.82	86,33	n/c
Maximum PSNR [dB]	50	48	47	47	48	40	50.18	49.58	52.78	57.59	65.44	40.39	94,21	n/c
Standard Deviation [dB]	1.37	1.28	1.24	1.32	1.26	2.43	3.11	2.75	2.77	2.32	3.51	2.55	0.68	n/c

Table 2 Test Set A: Averages of Bit-Error Rates (RBER being raw error correction), Watermark Detection Rates, as well as Average Minimum, Maximum and Standard Deviation of PSNR, for Block-Luminance Watermark (BL), Dual-Domain-DFT (DDD) and Wet Paper Code / Blue channel embedding (WPC) at four different compression rates. n/c denotes values which have not been determined due to a Watermark Detection rate of 0% and J<x> denotes JPEG compression with a quality factor of <x> percent after embedding.

Algorithm:	Block-Luminance						DDD						Wet Paper Code	
Compression after Emb.	Raw	J100	J90	J75	J50	J25	Raw	J100	J90	J75	J50	J25	Raw	J100
Average RBER [%]	0.03	0.04	0.06	0.18	0.46	2.63	0.20	0.20	2.25	22.81	39.06	50.77	n/c	n/c
Average CBER [%]	0.01	0.01	0.01	0.02	0.05	0.62	0.00	0.00	0.49	22.23	36.43	44.32	n/c	n/c
WM Detection Rate [%]	99.33	99.33	99.33	98.67	95.85	80.58	98.59	98.59	20.42	0.00	0.00	0.00	98.59	0
Average PSNR [dB]	49.2	48.5	41.4	39.5	37.6	35.7	45.93	45.43	40.42	39.00	37.62	35.80	90.68	n/c
Minimum PSNR [dB]	46.6	46.3	39.5	35.9	34.3	32.2	38.61	38.52	36.52	34.83	33.94	32.15	89.62	n/c
Maximum PSNR [dB]	50.9	50.0	45.3	46.7	40.6	39.2	52.97	51.74	46.15	43.90	41.16	39.44	91.96	n/c
Standard Deviation [dB]	1.1	1.0	1.8	2.9	2.3	2.5	4.18	3.86	2.23	2.54	2.50	2.56	0.59	n/c

Table 3 Test Set B: Averages of Bit-Error Rates (RBER being raw and CBER after error correction), Watermark Detection Rates, as well as Average Minimum, Maximum and Standard Deviation of PSNR, for Block-Luminance Watermark (BL), Dual-Domain-DFT (DDD) and Wet Paper Code / Blue channel embedding (WPC) at four different compression rates. n/c denotes values which have not been determined due to a Watermark Detection rate of 0% and J<x> denotes JPEG compression with a quality factor of <x> percent after embedding.

For a comparative overview between the error characteristics of the different algorithms, the following diagrams illustrate the observed function of (raw) BER as function of PSNR in Figure 10 and Figure 11. For the block luminance algorithm, all 6 measurements are included in the graphs (green symbols). However, for the hierarchical DDD algorithm only the first three measurements have been visualized because the last three values are too large for the chosen scale and only one single measurement is included for the Wet Paper Code / Blue Channel LSB algorithm (blue symbol), due to the above mentioned non-robustness to JPEG compression.

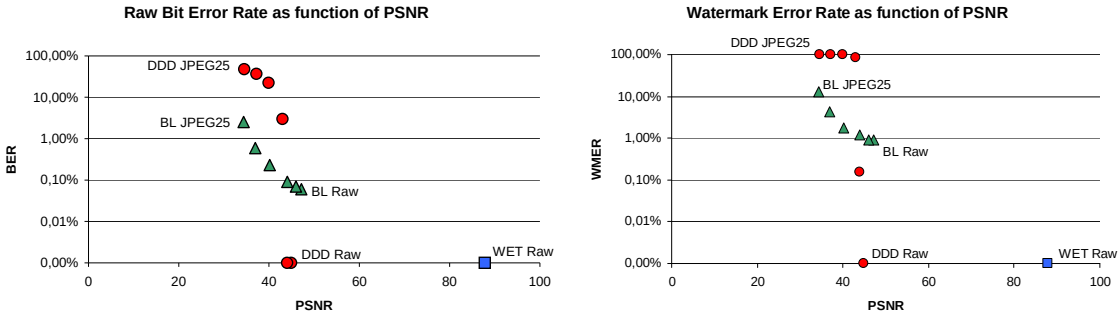

Figure 10 Bit Error Rates (left) and Watermark Error Rates (right) at different compression levels as function of PSNR for Block-Luminance (BL) Algorithm, Dual-Domain-DFT (DDD) and Wet Paper Code / Blue Channel LSB algorithms (WET) for Test Set A.

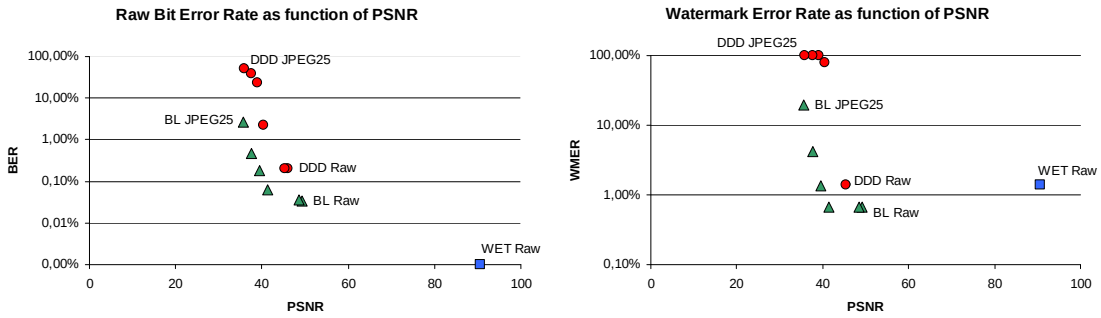

Figure 11 Bit Error Rates (left) and Watermark Error Rates (right) at different compression levels as function of PSNR for Block-Luminance (BL) Algorithm, Dual-Domain-DFT (DDD) and Wet Paper Code / Blue Channel LSB algorithms (WET) for Test Set B.

Regarding the cropping tests during the second test setup we analyzed the detection rates from the cropped images, considering how many annotations were completely or partially inside the cropping area and how many were cut out completely. The left illustration in Figure 12 exemplifies the three categories for objects after cropping. All objects that are completely located inside the cropping region (highlighted area in the center of the illustration) are denoted by *(c)*, partially cut objects by *(p)* and objects completely outside the cropping region are identified by *(o)*. The screenshot on the left-hand side of Figure 12 shows one example from our database *Test Set B*.

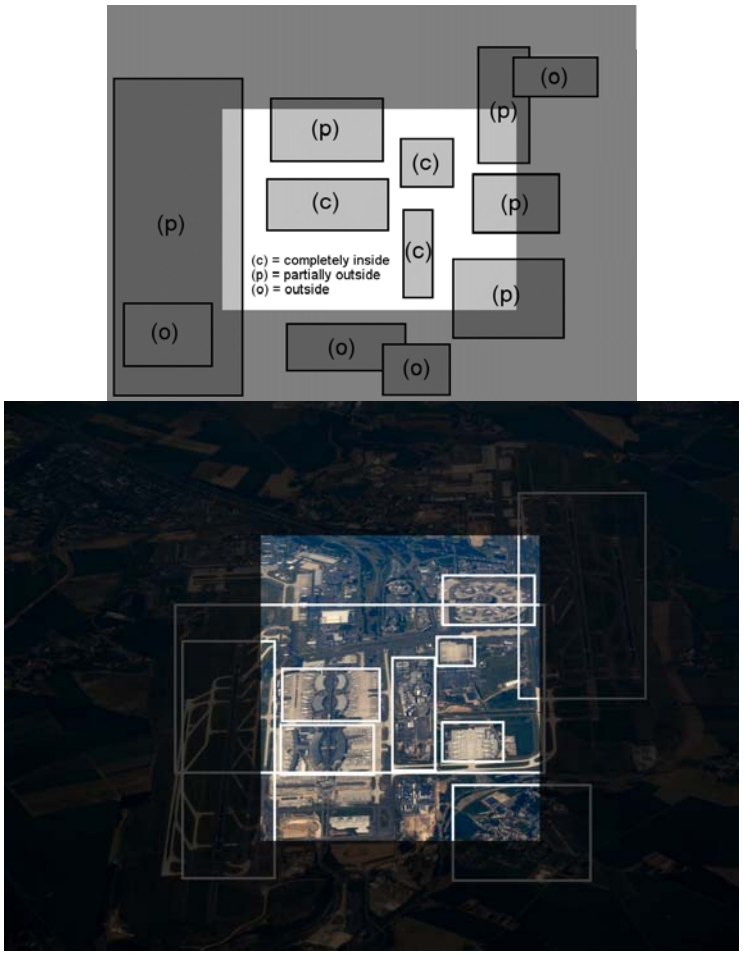

Figure 12 Cropping categories for objects: illustration of the three categories (left) and example cropping from Test Set B (right).

Table 4 provides a summary of the detection results for cropped objects from test set B.

Algorithm:	Block-Luminance				DDD				Wet Paper Code
	s=1	s=3	s=5	s=10	s=1	s=5	s=10	s=20	
Detected compl. obj. [%]	78.47	100.00	100.00	100.00	98.67	98.67	98.67	98.67	98.67
Detected partial obj. [%]	17.86	25.24	25.24	25.24	10.71	10.71	10.71	10.71	0.00

Table 4 Object annotations found after cropping areas from the source images

In the tests for the Block-Luminance algorithm, all objects that resided completely within the selected area could also be detected after cropping, given that an embedding strength *s* of 3 or higher was used. On the test-images created with *s=1* a few complete objects could not be detected after cropping. However, most of these could not be detected on the non-cropped image as well, so the low embedding strength factor seems to be the main problem here. With respect to partially contained objects, between 17% (*s=1*) and 25% (*s=3, 5, 10*) of them could be detected even after cropping. Objects that survived truncation had always been clipped at their bottom or at the right side, where not necessarily essential information is stored.
In the cropping tests with the Hierarchical DDD algorithm, the results were similar to the cropping tests of the Block luminance algorithm: Most complete objects could also be detected after cropping (or their detection failed on the respective non-cropped image as well) and a few partially truncated objects could still be detected (this time all affected objects had been clipped at the right side only).
As the cropping tests with the Wet Paper Code algorithm showed, this is the algorithm that is most sensitive to cropping: Objects which were just partially included in the cropping area became not detected at all. On the other hand, objects that were completely inside the crop-

ping area became detected with the same reliability than on the non-cropped image: In our 15 reference images there was only one such object that could not be detected, but this effect was the same on the non-cropped image. The test results are discussed in the following subsections separately for each algorithm.

Block-Luminance Watermark:

Our experiments have shown that this scheme is relatively robust to JPEG compressions up to 50% with an overall BER of around 0.5% and a watermark detection rate of roughly 95.8% as to be seen from Table 2 and Table 3. This performance could be observed both for cropped and non-cropped images. While the ratio between BER and Watermark Detection Rate may be improved by better error correction codes (in this evaluation, this was simply performed by triple redundancy), this embedding approach has two limitations: firstly, due to the nature of this correlation approach, there is no a-priori guarantee with respect to the success of any embedding attempt. In practice, this may lead to an expected failure of successful retrieval of the object watermarks in approximately 0.9% of cases, even without any compression applied to the watermarked images. Secondly, due to block-based scheme, capacity is limited to one pixel per block. In our implementation, with a 15-bit synchronization pattern for blind detection, this limits the embedding region to a minimum width of 120 pixels.
With average PSNR of 47.16 dB in test set A and 49.2 dB in test set B, the measurable transparency can be considered relatively high,

Wet Paper Codes:

As discussed in section 3, Wet Paper codes are mainly used for steganographic schemes and thus robustness against format conversion or other attacks is not a design goal for this embedding technique. This has obviously been confirmed by our experiments, where none of the watermarks could be retrieved after compression at any rate. On the other hand, our embedding scheme, having matrix parameters of $q=8$ and $n=40$, allows for relatively higher capacity embedding. For our protocol, this results in a relaxation of the minimum width limitation of the embedding are to 40 pixels as compared to 120 pixels for the block luminance scheme. Finally, the transparency in terms of PSNR is in the order of three magnitudes higher than for the first method (in average 87,82 dB for test set A and 90.68 for test set B). Although undoubtfully, the interpretation of PSNR as a transparency measurement involves some uncertainty, our complementary subjective tests have shown no practical visibility of the watermarks.

Hierarchical DDD algorithm:

Compared with the Block Luminance algorithm, the Hierarchical DDD scheme is less robust against JPEG compression. While JPEG100 compression behaves similarly compared with the uncompressed image in terms of bit and watermark error rates, higher compression leads to a clear increase of these error rates. Already at JPEG75, too many bit errors occur to allow error corrections. The possibilities to enhance this situation by increasing the embedding strength are very limited: At an embedding strength of $s=20$ the PSNR value of the uncompressed image already sinks falls 40 dB, where about 50% of the watermarks survive JPEG75 compression now. However, at JPEG50 again no more watermarks can be retrieved correctly. On the other hand, using the Block Luminance scheme even with an embedding strength of $s=3$ about 80% of the watermarks survived a JPEG25 compression. After increasing the Embedding strength to $s=10$ the PSNR value was still above 40 dB and all watermarks could still be detected even after this compression rate.
Looking at the changes introduced by the embedding process, the PSNR value range is a bit lower than with the Block Luminance algorithm. Depending on the embedding strength, we measured average PSNR values of the uncompressed output images between 53.3 and 41.1

dB using the Block Luminance algorithm. With the DDD algorithm these values were between 46.3 and 39.8 dB. However, for a human viewer the changes introduced by the Block Luminance algorithm are a bit more noticeable, especially on higher embedding strengths. Since the DDD algorithm embeds the watermark in the frequency domain, the introduced changes are spread more uniformly throughout the entire object so they are less obvious for the human user. With respect to capacity, the DDD scheme provides a relatively low capacity for the class inheritance in the magnitude domain. In our parameterization, only a maximum of $l=64$ hierarchical classes can be embedded in 16×16 pixel blocks ($n=16$). Thus the payload per block yield 6 bits. Similarly, the capacity for instantiation data in the second domain, the phase coefficients, is limited to 4 bits payload before error correction. Although in comparison to BL, capacity is not significantly higher than 1 pixel per block, however one major advantage of DDD is the fact that class hierarchies can be restored even from one single block only. In our setup cropped areas between 16×16 and 32×32 of pixel size are sufficient to reconstruct the class hierarchy.

5. Personnel Supported

Prof. Jana Dittmann (jana.dittmann@iti.cs.Uni-Magdeburg.de)
Dr. Claus Vielhauer (claus.vielhauer@iti.cs.uni-magdeburg.de)
Maik Schott (mschott@cs.uni-magdeburg.de)
Milen Touchev (Milen.Touchev@Student.Uni-Magdeburg.de)
Tobias Scheidat (tobias.scheidat@iti.cs.uni-magdeburg.de)
Tobias Hoppe (tobias.hoppe@iti.cs.uni-magdeburg.de)

6. Publications

[ViSc2006] C. Vielhauer and M. Schott, Image Annotation Watermarking: Nested Object Embedding using Hypergraph Model, Proceeding of the 8th ACM Workshop on Multimedia and Security, pp. 182-189, Geneva, Switzerland, 2006

[ViDi2007] C. Vielhauer and J. Dittmann, Nested Object Watermarking: Comparison of Block-Luminance and Blue Channel LSB Wet Paper Code Image Watermarking, accepted for publication in: Proceedings of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX, 2007

7. Interactions/Transitions

Interaction with a project from German Science Foundation IlluWaz (Illustration Watermarking) in the field of BlobContours: Thomas Vogel thomas.vogel@iti.cs.uni-magdeburg.de. The developed sources will be used in the project for automated object contour selection.

8. New discoveries, inventions, or patent disclosures.

none

9. Honors/Awards

none

10. References

[1] Borland Delphi Developer Network: http://bdn.borland.com/delphi/, 2006
[2] WordNet – Princeton University Cognitive Science Laboratory, http://wordnet.princeton.edu/, 2005

[3] Christiane Fellbaum; WordNet: An Electronic Lexical Database (Language, Speech and Communication), MIT Press, ISBN 0-262-06197-X, 1998.
[4] T. Vogel, D. Q. Nguyen, J. Dittmann, BlobContours: Adapting Blobworld for Supervised Color- and Texture-Based Image Segmentation. SPIE Electronic Imaging - Multimedia Content Analysis, Management, and Retrieval 2006, San Jose, California (USA), 15-19 January
[5] A. Nikolaidis and I. Pitas, Region based image watermarking, IEEE Transactions on Image Processing, Vol. 10, Issue 11, pp. 1726-1740, 2001
[6] Jana Dittmann, *Digitale Wasserzeichen. Grundlagen, Verfahren, Anwendungsgebiete*, Springer, Berlin, 2000, ISBN 3-540-66661-3
[7] J. Fridrich: *Methods for data hiding*, Center for Intelligent Systems & Department of Systems Science and Industrial Engineering, SUNY Binghamton
[8] J. Fridrich, M. Goljan, P. Lisonek, and D. Soukal, Writing on Wet Paper, with IEEE Trans. on Sig. Proc., Special Issue on Media Security, Eds. T. Kalker and P. Moulin, vol. 53, pp. 3923-3935, October 2005.
[9] J. Dittmann: Digitale Wasserzeichen. Grundlagen, Verfahren, Anwendungsgebiete, Springer, Berlin, 2000 (in German)
[10] C. Vielhauer and M. Schott, Image Annotation Watermarking: Nested Object Embedding using Hypergraph Model, Proceeding of the 8th ACM Workshop on Multimedia and Security, pp. 182-189, Geneva, Switzerland, 2006
[11] J.Fridrich, M. Goljan, P. Lisoněk and D. Soukal: Writing on Wet Paper, In: E. J. Delp III, P. W. Wong (Eds.): Security, Steganography, and Watermarking of Multimedia Contents VII. Proceedings of SPIE, San Jose, CA, 2005.
[12] J. Fridrich, M. Goljan and D. Soukal: Perturbed Quantization Steganography with Wet Paper Codes; In Proceedings of ACM Multimedia Workshop, Magdeburg, Germany, September 20-21, pp. 4-15, 2004.
[13] J. Fridrich, M. Goljan and D. Soukal: Efficient Wet Paper Codes, Information Hiding, 7th International Workshop, LNCS vol. 3727, Springer-Verlag, New York, pp. 204-218, 2005.
[14] I. Cox, J. Kilian, T. Leighton, T. Shamoon, Circularly symmetric watermark embedding in 2-{D} {DFT} domain, IEEE Transactions on Image Processing, Vol. 6, No. 12, pp. 1673-1687, 1997
[15] V. Solachidis and I. Pitas, Circularly symmetric watermark embedding in 2-D DFT domain, Proceedings of the 1999 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing , Vol. 6, pp. 3469-3472, 1999
[16] E. Ganic, S.D. Dexter, A.M. Eskicioglu, Embedding multiple watermarks in the DFT domain using low- and high-frequency bands, In Proceedings of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VII, Vol. 5681, pp. 175-184, 2005

Annexes

A) Paper hardcopies of [ViSc2006] and a working draft of [ViDi2007]

B) AnnoWaNo User Manual (Manual.doc)

C) Reference annotations (Reference Annotations.doc)

D) Binary resources (on DVD only)

1. AnnoWaNo Latest Version Maik
2. AnnoWaNo Source Code Maik
3. Structured overview on all test results (Testresults.xls)
4. Complete raw log files (logfiles.zip)